(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **10154078 A**

(43) Date of publication of application: **09.06.98**

(51) Int. Cl **G06F 9/445**
G06F 13/00

(21) Application number: **09265696**

(22) Date of filing: **30.09.97**

(30) Priority: **07.10.96 US 96 722434**

(71) Applicant: **INTERNATL BUSINESS MACH CORP <IBM>**

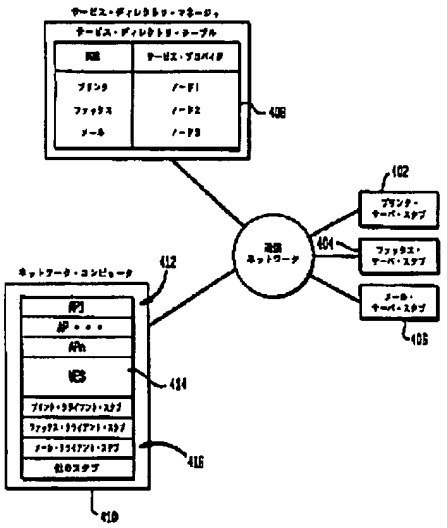
(72) Inventor: **MARCY DEBARAKONDA**
AJAI MOOINDORA
DEBORAH JEAN ZHUKOVSKI

(54) **ACCESS SUPPLY METHOD FOR NETWORK SERVICE**

(57) Abstract:

PROBLEM TO BE SOLVED: To realize the service of a network base by detecting requested remote network service and down-loading an object into the memory of a network node.

SOLUTION: A service directory table(SDT) 408 contains information on three service providers. Various applications 412, the action virtual environment supervisor(VES) object 414 containing the table of constituted service and a passive stub object 416 supplying connection to service are contained in the specified instance of a network computer 410. When the application requests access to remote network service, the remote service network is detected, the object is down-loaded into the memory of the network node and the application executes access when remote network service is requested.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－154078

(43)公開日　平成10年(1998)６月９日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　9/445		Ｇ０６Ｆ　9/06	４２０Ｊ
13/00	３５７	13/00	３５７Ｚ

審査請求　未請求　請求項の数20　ＯＬ　（全 13 頁）

(21)出願番号　特願平9－265696

(22)出願日　平成９年(1997)９月30日

(31)優先権主張番号　０８／７２２４３４

(32)優先日　1996年10月７日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　マーシー・デバラコンダ
アメリカ合衆国10510、ニューヨーク州ブライアークリフ・マナー、キウェイス・カーブ　611

(74)代理人　弁理士　坂口　博　（外１名）

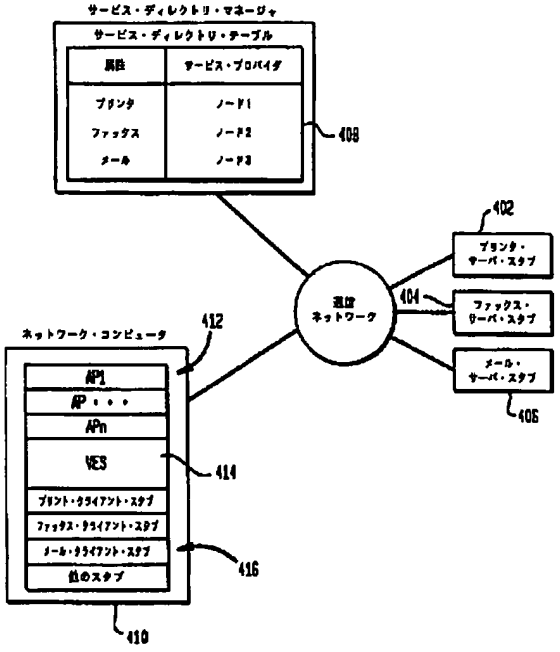
最終頁に続く

(54)【発明の名称】　ネットワーク・サービスへのアクセス提供方法

(57)【要約】

【課題】　クライアントのために、ネットワーク・ベースのサービスを可能にする、柔軟で強力且つ移植性のある手段を提供する。

【解決手段】　ネットワーク・コンピュータ上で実行されるアプリケーションに、ネットワーク・サービスへのアクセスを提供する、ＶＥＭと呼ばれるダウンロード可能な即時ミドルウェアのためのシステム及び方法が提供される。ネットワーク・サービスには、印刷及びローカル記憶などのシステム・サービスが含まれる。ＶＥＭがデフォルト指定のクライアント・サービスを構成し、これらのサービスに関する情報を記憶する。ネットワーク・コンピュータ上で実行されるアプリケーションがサービスの１つを使用したいとき、これはそのローカルＶＥＭと通信する。ＶＥＭはハンドルを適切なサービスに返却し、サービス要求を完了する。



1

【特許請求の範囲】

【請求項１】ネットワーク・クライアント上で実行されるアプリケーションに、ネットワーク・サービスへのアクセスを提供する方法であって、

前記アプリケーションがリモート・ネットワーク・サービスへのアクセスを要求する時を判断するステップと、

前記アプリケーションが前記リモート・ネットワーク・サービスへのアクセスを要求したとの判断に応答して、前記アプリケーションにより要求された前記リモート・ネットワーク・サービスを突き止めるステップと、

少なくとも１つのオブジェクトをネットワーク・ノードのメモリにダウンロードするステップと、

を含み、前記アプリケーションが前記リモート・ネットワーク・サービスを要求時にアクセスすることを可能にする方法。

【請求項２】前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・スタブをダウンロードするステップを含む、請求項１記載の方法。

【請求項３】前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・ハンドルをダウンロードするステップを含む、請求項１記載の方法。

【請求項４】前記ネットワーク・クライアントのユーザに、以前にアクセスが許可された使用可能なネットワーク・サービスのビジュアル表示を提供するステップを含む、請求項１記載の方法。

【請求項５】前記ネットワーク・クライアントがネットワーク・コンピュータである、請求項１記載の方法。

【請求項６】前記ネットワーク・ノードがＪＡＶＡ端末である、請求項１記載の方法。

【請求項７】前記ネットワーク・ノードがパーソナル・デジタル・アシスタント（ＰＤＡ）である、請求項１記載の方法。

【請求項８】前記ネットワーク・サービスがシステム・サービスである、請求項１記載の方法。

【請求項９】前記システム・サービスが印刷サービスである、請求項８記載の方法。

【請求項１０】前記システム・サービスが永久記憶である、請求項８記載の方法。

【請求項１１】前記システム・サービスがシステム・モニタリングである、請求項８記載の方法。

【請求項１２】前記システム・サービスがシステム管理である、請求項８記載の方法。

【請求項１３】仮想環境マネージャが前記ネットワーク・サービスへのアクセスを提供する、請求項１記載の方法。

【請求項１４】前記仮想環境マネージャが、前記ネットワーク・ノード上で実体化される仮想マシン上で実行される、請求項１３記載の方法。

2

【請求項１５】前記ネットワーク・サービスへのアクセスがもはや必要とされない時を判断し、前記ネットワーク・ノードの前記メモリから前記オブジェクトの除去を開始するステップを含む、請求項１記載の方法。

【請求項１６】前記リモート・サービス・プロバイダにより、前記ネットワーク・クライアントが使用可能なサービスのセットを、第３者ディレクトリに登録するステップを含み、前記オブジェクトが前記ディレクトリを参照して、前記サービスへのアクセスを獲得する、請求項１記載の方法。

【請求項１７】前記ネットワーク・ノード上で実行される複数のアプリケーションにより、前記サービスへのアクセスを共用するステップを含む、請求項１記載の方法。

【請求項１８】あるアプリケーションにより必要とされるサービスへのアクセスを、前記アプリケーションにのみ知れる固有のキーによってのみ提供することにより、前記サービスを他のアプリケーションから分離するステップを含む、請求項１記載の方法。

【請求項１９】ネットワーク・コンピュータ上で実行されるアプリケーションに、システム・サービスへのオンデマンド・アクセスを提供する方法であって、

ａ）ユーザからの要求に応答して、リモート・ネットワーク・サービスへのアクセスを提供する仮想環境マネージャをダウンロードするステップと、

ｂ）システム・サービスのセットを識別するステップと、

ｃ）適切なリモート・サービス・プロバイダに連絡し、各前記サービスに対するクライアント・スタブまたはハンドルをダウンロードすることにより、前記システム・サービスのセットを構成するステップと、

ｄ）構成済みサービスのリストを保持するステップと、

ｅ）前記構成済みサービスへのアクセスを提供するステップと、

を含む、方法。

【請求項２０】複数のアプリケーションにより、前記システム・サービスへのアクセスを共用するステップを含む、請求項１９記載の方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明はネットワーク・コンピュータに関する。より詳細には、本発明は、ネットワーク・サービス（例えば印刷及びローカル記憶などのシステム・サービス）へのアクセスを、ネットワークに接続されるコンピュータ上で実行されるアプリケーションに提供する方法及び手段に関する。

【０００２】

【従来の技術】ネットワーク・コンピュータのパラダイムが普及すると、ネットワーク・コンピュータとして知られる、独立の存在のための手段を有さない単純で安価

なデスクトップ・コンピュータが一般的となる。こうした装置は（単純化のために）、低機能のオペレーティング・システムを実行し、ローカル記憶、印刷及びモニタリングなどの基本システム・サービスをサーバに頼り得る。ＰＣなどの従来のクライアントは、それら自身の基本システム・サービスを提供する一方で、ネットワークからも追加のサービスを求めがちである。

【０００３】これらの基本システム・サービスをアプリケーションに提供するために使用される既存のアプローチは、ネットワーク・コンピュータ上に完全機能の動作環境を存在させたり、或いは各アプリケーションが必要なサービスの自身のセットを提供することを要求した。前者のアプローチはネットワーク・クライアントにとって実現可能でない。なぜなら、これらのコンピュータは（例えばネットワーク・コンピュータの場合、）完全機能の動作環境をサポートするための、必ずしも十分な物理資源を装備していないからである。こうした物理資源には、物理メモリ及び接続周辺装置（例えばディスク・ドライブ）などが含まれる。後者のアプローチでは、各アプリケーションが必要な基本システム・サービスのサポートを提供できるように、それが実行されるプラットフォーム及び環境を意識しなければならない欠点を有する。更に前者のアプローチは、ネットワーク・クライアントのコストを追加し、後者のアプローチは、アプリケーションの設計の複雑度を増す。

【０００４】

【発明が解決しようとする課題】本発明の目的は、クライアントのために、ネットワーク・ベースのサービスを可能にする、柔軟で強力且つ移植性のある手段を提供することである。

【０００５】

【課題を解決するための手段】好適な実施例では、仮想環境マネージャ（ＶＥＭ）と呼ばれるダウンロード可能なミドルウェアが提供される。ＶＥＭはアプリケーションがクライアント・コンピュータ及びそれが接続されるサーバのアーキテクチャ及び環境に完全に独立に開発されることを可能にする。クライアント（すなわちネットワーク・コンピュータ）がサービスをアクセスするために、ＶＥＭは１つまたは複数の接続サーバ上で使用可能な、サービス・ディレクトリ・テーブルを照会する。サービス・ディレクトリ・テーブルへのアクセスは、指示されたサービス・プロバイダに接続するためのハンドルを返却する。

【０００６】

【発明の実施の形態】本発明と一緒に使用するのに好適な疎結合のシステムが、図１に示される。システムは、通信ネットワーク１１２により相互接続される複数のコンピュータ１０２乃至１０６を含む。これらのコンピュータの幾つかは、ネットワーク・コンピュータ１０６（１）、１０６（２）として知られる。なぜならこれらは、それらの基本機能の多くを提供するためにネットワークを介して使用可能なサービスに頼るからである。他のコンピュータは、サービス・プロバイダ１０４（１）、１０４（２）、１０４（３）として知られる。なぜなら、これらは基本システム・サービスなどのネットワーク・サービスをネットワーク・コンピュータに提供するからである。コンピュータの幾つかは、サービス・ディレクトリ・マネージャ（ＳＤＭ）１０２としても知られる。なぜならこれらは、サービス・プロバイダによりネットワーク・コンピュータに提供されるサービスのリストを保持するからである。

【０００７】ネットワーク・コンピュータ１０６が、例えばＪＡＶＡ端末、パーソナル・デジタル・アシスタント（ＰＤＡ）、またはインターネット端末上に実現され得る。通信プロトコルは、ＨＴＴＰ及びＴＣＰ／ＩＰである。通信ネットワーク１１２は、例えばトークン・リングである。サービス・システム１０２、１０４（１）、１０４（２）、１０４（３）は、例えばＡＩＸ４．２を用いるＩＢＭ　ＲＩＳＣ　Ｓｙｓｔｅｍ／６０００マシン上で実現され得る。

【０００８】論理構成が上述の物理システム上に配置され得る。この構成は、３つの階層のクライアント／サーバ機構により説明される。この機構では、階層１はクライアント機能を、階層２はサービス・プロバイダを、階層３はデータ・オブジェクト・サーバ（すなわち情報記憶装置）を表す。本実施例で述べられるシステムは、階層１及び階層２、並びにそれらの間のインタフェースに関連する。階層２と階層３との間のインタフェースは制限されず、従ってユーザは、従来の信頼の置けるデータ情報アクセス・システムに頼ることができる。サービス・プロバイダは、完全に階層２サーバ上に存在するか、或いは特殊なサービス・スタブを提供することにより、その機能を階層１及び階層２に割当てるように設計され得る。

【０００９】各ネットワーク・コンピュータは、仮想環境マネージャ（ＶＥＭ）１０８（１）、１０８（２）を含み、これらは通信ネットワーク１１２を通じてサービス・プロバイダからダウンロードされる。ＶＥＭはネットワーク・コンピュータのランダム・アクセス・メモリ（図示せず）内に実体化されるプログラム・コードとして実現される。特にＶＥＭは（"オブジェクト指向プログラミング"の意味において、）VEMCLASS（表１で定義される）と呼ばれる基本クラスを提供し、これに頼って、オブジェクトの集合を定義する。各ＶＥＭに含まれるオブジェクトの１つは、仮想環境スーパバイザ（ＶＥＳ）１０９（１）、１０９（２）と呼ばれる活動エンティティであり、これはＶＥＭ状態を保持する役割をする。好適な実施例では、ＶＥＳは活動ＪＡＶＡオブジェクトであり、VEMCLASSから直接実体化される。ＶＥＭ状態は、構成済みサービスのテーブル１１０（１）、１１

０（２）、及び活動アプリケーションのテーブル１１２（１）、１１２（２）を含む。構成済みサービスのテーブル及び活動アプリケーションのテーブルの両者も、各ネットワーク・コンピュータのランダム・アクセス・メモリ内で実体化される。アプリケーションはＶＥＳと対話するためにVEMCLASSから継承する。オブジェクト・クラスはアプリケーションにより編成され、アプリケーションのネーム空間に結合される。

【００１０】本発明の実施例によれば、ネットワーク・コンピュータがスイッチ・オンされるとき、これはブート・プロセスを実行し、自身を使用可能な状態に準備する。ブート・プロセスの終了時にネットワーク・コンピュータがユーザに識別を催促する。識別プロセスは、例えばユーザ名及びパスワードをタイプ入力するなどである。ユーザ識別の検証の後、ネットワーク・コンピュータはユーザにデスクトップ環境を提供する。

【００１１】デスクトップ環境は、好適にはＪＡＶＡコンテナである。ＪＡＶＡコンテナはＪＡＶＡ仮想マシン上で実行される。コンテナ及びＪＡＶＡ仮想マシンの両者は、ネットワーク・コンピュータ上で実行されるウェブ・ブラウザ１１４（１）、１１４（２）（例えばネットスケープ・ナビゲータ）などのプログラムにより提供される。ブラウザはネットワーク・コンピュータのブート・シーケンスの間にダウンロードされるか、或いはネットワーク・コンピュータの統合部分として提供され得る。

【００１２】初期化プロセスの一部として、ブラウザはユーザのためにＶＥＳ及び構成ファイルをサービス・プロバイダからダウンロードする。ＶＥＳは構成ファイルを用いて、ユーザのデスクトップ環境（デスクトップ）をセットアップする。デスクトップの例（ホームページの形式を取り得る）が図６に示される。構成ファイルは、システム及びユーザが最も頻繁にアクセスするアプリケーション・サービスのリストを含む。これらのシステム及びアプリケーション・サービスは、デスクトップ上に表示されるアイコン６０２またはボタン６０４、及び（または）テキスト・リスト６０６として実現され得る。システム及びアプリケーション・サービスはハイパリンクとして、または適切なＪＡＶＡコードがダウンロード済みの直接制御ボタンとしてアクセスされ得る。ＶＥＭの残りは、ユーザ・アプリケーションと一緒にダウンロードされる。

【００１３】ホーム・ページの初期ロギング及びダウンロードが図２に示される。ステップ２０２で、ネットワーク・コンピュータがユーザにより、パワーオンまたはリセットされる。それに応答してステップ２０４で、ネットワーク・コンピュータが、そのマシン特有の従来のブート・シーケンスを開始する。このシーケンスは、ウェブ・ブラウザのダウンロードを含むように拡張され得る。ステップ２０６で、ネットワーク・コンピュータがユーザにユーザ名及びパスワードを催促する。ネットワーク・コンピュータはステップ２０８で、従来方法によりユーザ名及びパスワードが有効か否かを判断する。当業者には理解されるように、ステップ２０６及び２０８は、代わりにステップ２０４の一部としても実行され得る。ステップ２０８で、ユーザ名またはパスワードのいずれかが有効でないと判断されると、本方法はステップ２０６に戻る。ユーザ名及びパスワードが有効な場合には、ステップ２１０で、ネットワーク・コンピュータがＶＥＳ及び構成ファイルをダウンロードする。

【００１４】ＶＥＳがユーザのためにダウンロードされると、これは構成ファイル内に存在する全てのサービスの構成を開始する。構成ファイルは従来のフラット・ファイル形式でよい。例えば構成ファイルは、ネットスケープ・ナビゲータ３．０により使用されるタイプのブックマーク・ファイルの形式である。この構成は、適切なサービス・プロバイダに連絡して、クライアントのサービス・テーブル１１０をサーバ情報により初期化することにより達成される。スタブもまた、階層１及び２上に存在するサービスのためにダウンロードされる。

【００１５】現構成済みサービスに加え、ＶＥＳはネットワーク上で使用可能な全てのサービスへのアクセスを有する。このことは、ユーザが任意の使用可能なサービスをデスクトップに追加し、結果的に、ユーザの構成ファイルに追加することを可能にする。

【００１６】サービス・アクセスが、サービス・ディレクトリ・マネージャ（ＳＤＭ）により提供される。ＳＤＭは、サービス・ディレクトリ・テーブル（ＳＤＴ）（図１参照）として参照されるサービスをテーブルを保持する。サービス・ディレクトリ・テーブルは、システム及びネットワーク上のサービス・プロバイダにより提供されるアプリケーション・サービスに関する情報を含む。ネットワーク上に複数のＳＤＭが存在し得り、各々がＳＤＴを保持するか、ＳＤＴの１つのインスタンスをアクセスする。

【００１７】サービス・プロバイダがネットワークに接続されるとき、これは自身がＳＤＭに提供するサービスのセットを公表する。これらのサービスのセットを公表するにおいて、各サービス・プロバイダにより実行されるステップが、図３に示される。ステップ３０２で、サービス・プロバイダは自身が提供するサービスのリストを生成する。ステップ３０４で、サービス・プロバイダはネットワーク内のサービスのリストを保持する責任を負うＳＤＭを識別する。ステップ３０６で、サービス・プロバイダはサービスのリストを走査し、次に登録されるべきサービスを決定する。登録されるべき別のサービスが存在する場合、ステップ３０８で、サービス・プロバイダは登録メッセージをＳＤＭ（ディレクトリＭ）に送信し、次にステップ３０６に戻る。リスト内にもはやサービスが存在しない場合には、ステップ３１０で初期

化が完了する。ここでステップ３０６及び３０８は、１つのステップにより置換され得ることが理解されよう。その場合、リスト全体が１度だけ読出され、次に適切なＳＤＭに１つのメッセージまたはメッセージ・シーケンスの一部として送信される。

【００１８】様々なクライアント・スタブがネットワーク・コンピュータにおいて配置されるシステム状態の例が、図４に示される。図４は３つのサービス・プロバイダ、すなわちプリント・サーバ４０２、ファックス・サーバ４０４、及びメール・サーバ４０６を示す。ＳＤＴ４０８は、３つのサービス・プロバイダに関する情報を含む。特に、情報はサービスのタイプ（属性）、及び特定のサービスを提供するサービス・プロバイダの位置を示す。ネットワーク・コンピュータ４１０の特定のインスタンスには、様々なアプリケーション４１２（ＡＰ１－ＡＰｎ）、構成済みサービスのテーブルを含む活動ＶＥＳオブジェクト４１４、及びサービスへの接続を提供する受動スタブ・オブジェクト４１６（プリント・クライアント・スタブ、ファックス・クライアント・スタブ、メール・クライアント・スタブ）が含まれる。ＶＥＭはＶＥＳ、スタブ、及び各アプリケーションのVEMCLASS部分を含む。

【００１９】次にＶＥＭについて詳述することにする。参考として、ＶＥＭのコード定義は、VEMCLASS、並びにクライアント及びサーバ・インタフェースを含む。これらは表１及び表２にそれぞれ示される。図５は、アプリケーションがネットワーク・サービスを使用したいときのフロー制御プロセスを示す。前述のように、クライアントへのＶＥＳのロードはブラウザにおいてＨＴＭＬページを見ることに類似である。ＶＥＳがダウンロードされ、そのinit()メソッドが呼び出される。init()メソッドは、クラス変数VE_supervisorが一度だけ初期化されるように同期する。ＶＥＳを開始するためのあらゆる続く試みが不能にされる。ＶＥＳはそれ自身をVEM_idインスタンス変数をセットすることにより、ｉｄ＝０に割当てる。クラス変数num_AFEが次に増分される。レジストリ（構成済みサービス・テーブル１１０及びアプリケーションのテーブル１１２を含む）、及び共用サービス・テーブルが作成される。最後に、ディレクトリ・サービス・リモート・オブジェクトが、ｈｔｍｌファイルに渡されるＤＳサーバ・パラメータを用いて、実体化される。

【００２０】

【表１】

```
/*ＶＥＳが実体化し、アプリケーションが継承するＶＥＭ基本クラスを下記に示す。*/
public class VEMCLASS extends Applet implements VEMBaseInterface {
/*VE_supervisorは、ＶＥＳ及び全てのアプリケーションが使用可能な単一の変数*/
private static VEMCLASS VE_supervisor = null;
/*num_AFEは、アプリケーションＩＤを生成するために使用される単一の単調増加のカウント*/
private static int num_AFE = 0;
/*VEM_idは、アプリケーションのＩＤの記憶を提供する、アプリケーション当たりの変数*/
protected int VEM_id;
/*レジストリは、構成済みサービス・テーブル及びアプリケーション状態の記憶を提供する。*/
private Hashtable registry;
/*shared_service変数は、共用サービスの迅速な探索機構を提供する。*/
private Hashtable shared_services;
/*String primary,secondary DS_Server変数は、ＳＤＭの位置を記憶する。*/
private String primary_DS_server, secondary_DS_server;
/*directory変数はＳＤＭへの直接アクセスを提供する。*/
private ServDir directory;
public VEMCLASS() {
}
public void init() {
}
/*表２に記述されるVEMBaseInterfaceの実現*/
public Cookie registerAFE(String name)
throws VEMRegistryException {
}
public Cookie registerService(String name)
throws VEMRegistryException {
}
public Cookie registerService(String name,String [] attrs)
throws VEMRegistryException {
}
public Cookie registerService(String name,boolean shared)
throws VEMRegistryException {
}
public Cookie registerService(String name,String [] attrs,boolean shared)
throws VEMRegistryException {
}
public ServiceHandle getServiceHandle(Cookie c)
throws VEMRegistryException {
}
public Object getServiceInstance(Cookie c)
throws VEMRegistryException {
}
public boolean revokeService(Cookie c)
throws VEMRegistryException {
```

```
}
/*ＶＥＳ状態を直接管理する内部ルーチン*/
synchronized private void addToRegistry(Cookie c,
RegistryElement r)
throws VEMRegistryException {
}
synchronized private RegistryElement getFromRegist
ry(Cookie c,) {
}
synchronized private String removeFromRegistry(Coo
kie c,)
throws VEMRegistryException {
}
synchronized private Cookie shareService(int o,Str
ing name) {
}
synchronized private void removeIfShared(int o,Str
ing name) {}
}
```

【００２１】

【表２】

```
/*ＶＥＭ基本インタフェース*/
public interface VEMBaseInterface {
public static final boolean SHARED = true;
/*VEMSupervisorに、常駐アプリケーション・フロント
・エンドを通知する。アプリケーションの任意のＶＥＭ
状態情報への、アプリケーション及びVEMSupervisorだ
けのアクセスを許可するために、固有キーが返送され
る。*/
/*アプリケーション・フロント・エンド（ＡＦＥ）をＶ
ＥＭに登録する。*/
public Cookie registerAFE(String name)
throws VEMRegistryException ;
/*VEMSupervisorに、ＡＦＥにより必要とされる要求サ
ービスを通知するメソッド。この呼び出しは、ＶＥＭ状
態をＡＦＥが動的にアクセス可能なサービスへのハンド
ルにより更新済みである。ＡＦＥがハンドルを用いて、
サービスとの通信をセットアップする。固有キーが返送
され、サービス・ハンドルへのＡＦＥ排他アクセスを可
能にする。*/
/*ＶＥＭ状態を、名前"name"を有する任意のサービスの
ためのハンドルにより、セットアップする。*/
public Cookie registerService (String name)
throws VEMRegistryException;
/*ＶＥＭ状態を、名前"name"を有し、属性"attrs"を含
むサービスのためのハンドルにより、セットアップす
る。*/
public Cookie registerService(String name,Strin
g [] attrs)
throws VEMRegistryException;
/*共用が"SHARED"に等しい、すなわち真ならば、名前"n
ame"を有するサービスがＶＥＭに知れているか否かを確
認する。知れていないならば、ＶＥＭ状態を共用サービ
スによりセットアップし、所有者を"この"ＡＦＥにセッ
トする。関連キーを返却する。*/
public Cookie registerService(String name, boolean
 shared)
throws VEMRegistryException;
/*共用が"SHARED"に等しい、すなわち真ならば、名前"n
ame"及び属性"attrs"を有するサービスがＶＥＭに知れ
ているか否かを確認する。知れていないならば、ＶＥＭ
状態を共用サービスによりセットアップし、所有者を"
この"ＡＦＥにセットする。関連キーを返却する。*/
public Cookie registerService(String name,Strin
g [] attrs, boolean shared)
throws VEMRegistryException;
/*ＡＦＥがＶＥＭに知れるサービスへのアクセスを獲得
するメソッド*/
/*サービスとの接続をセットアップするために使用され
得る、そのサービスに関連付けられるハンドルを獲得す
る。*/
public ServiceHandle getService(Cookie c)
throws VEMRegistryException;
/*しばしば共用サービスとして使用されるサービス・ス
タブへのアクセスを獲得する。*/
public Object getServiceStub(Cookie c)
throws VEMRegistryException;
/*全てのＶＥＭ状態からサービスを除去するメソッド*/
/*キーにより識別されるサービスを除去する。サービス
が共用される場合、所有者ＡＦＥだけがそれの除去を許
可される。*/
public boolean revokeService(Cookie c)
throws VEMRegistryException;
}
/*ＳＤＭインタフェース*/
public interface VEM_ServDir extends java.rmi.Remo
te {
/*サービスを探索するメソッド。"失敗"とマークされる
場合、サービスに対してハンドルが返却される。クライ
アントはそれをどのように処理するかを判断する。返却
されるハンドルは、{<ip_address>:<port #>:<flag>}で
あろう。フラグは"失敗"または"活動中"であろう。*/
/*サービス名に一致する最初のサービスに対するサービ
ス・ハンドルを返却する。*/
String lookUpService(String ServiceName)
throws java.rmi.RemoteException;
/*サービス名及び全ての属性に一致する最初のサービス
に対するサービス・ハンドルを返却する。*/
String lookUpService(String ServiceName,String []
ServiceAttr)
throws java.rmi.RemoteException;
```

```
/*サービス名に一致し、提供されたサービス・ハンドルの"後"に来る"次の"サービスに対するサービス・ハンドルを返却する。*/
String lookUpService(String ServiceName,String ServiceHandle)
throws java.rmi.RemoteException;
/*サービス名及び全ての属性に一致し、提供されたサービス・ハンドルの"後"に来る"次の"サービスに対するサービス・ハンドルを返却する。*/
String lookUpService(String ServiceName,String ServiceHandle,String [] ServiceAttr)
throws java.rmi.RemoteException;
/*サービス・カウントを獲得するメソッド。"失敗"とマークされる場合にもカウントされる。*/
/*一致するサービス名に対するカウントを返却する。*/
int getServiceCount(String ServiceName)
throws java.rmi.RemoteException;
/*一致するサービス名及び属性に対するカウントを返却する。*/
int getServiceCount(String ServiceName,String [] ServiceAttr)
throws java.rmi.RemoteException;
/*サービスに関するメタ情報を返却するメソッド。メタ情報は"活動中"サービスと同様、"失敗"サービスを含む。*/
/*ストリングの配列を返却する。第１の要素はサービスの名前であり、残りは属性である。入力はサービス・ハンドルである。*/
String [] getServiceNameAttr(String ServiceHandle)
throws java.rmi.RemoteException;
/*ストリングの配列を返却する。その配列の各要素は、特殊な区切り文字"$"により分離されるサービスの属性を含む。ここでサービス名、属性及びサービス・ハンドルが、"$"を含まないように規定する。*/
String [] getServiceNameAttr(String ServiceName)
throws java.rmi.RemoteException;
/*ストリングの配列を返却する。その配列の各要素は、最初にサービス名を、続いてサービス属性を含む。全ては"$"文字により区切られる。*/
String [] getListOfServices()
throws java.rmi.RemoteException;
}
```

【００２２】ＶＥＭがダウンロードされ、初期化されるとアプリケーション（ＡＰ）がダウンロードされる。（ここでの議論では、ＡＰがアプレットと仮定する。ＶＥＭはアプリケーションをサポートするが、アプレットは改良済みの管理能力に対して好適である。）そのinit()メソッドにおいて、ＡＰはregisterAFE()メソッドを呼び出すべきである。この呼び出しは、ＡＰのインスタンス変数VEM_idをクラス変数num_AFEにセットし、num_AFEを増分する。num_AFEは決して減分されることがないので、ＡＰはこの時点で固有のｉｄを有することになる。次に、レジストリ内にＡＰのエントリ（アプリケーションのテーブル１１２）を作成するために、ＶＥＳが呼び出される。この時点でエントリはアプレット・オブジェクト参照、ｉｄ及び名前だけを含むが、必要に応じて後に拡張され得る。registerAFE()メソッドは、レジストリ内のＡＰ情報へのアクセスが制御されるように、"クッキー（cookie）"と呼ばれる固有キーをＡＰに返却する。この時点で、ＡＰがＶＥＭ環境に完全に統合され、ＶＥＭ機能が実行可能になる。

【００２３】ＡＰはサービスを必要とするとき、最初にregisterService()メソッドの１つを用いて、ＶＥＭにサービスを登録するように要求する。サービス登録に際してＶＥＳはLookupService()メソッドを用いて、ディレクトリ・サービス（ＳＤＭ）をチェックする。このメソッドはサービスへのハンドルを獲得し、サービスがスタブを有するか否かを確認する。肯定の場合、スタブがダウンロードされる。ＶＥＭは次に、サービス・エントリをレジストリ（構成済みサービス・テーブル１１２）に追加する。レジストリはサービスのハンドル、及び適宜スタブの参照を含む。クッキーがＡＰに戻される。サービスがレジストリに追加されると、ＡＰ及びＶＥＳの両方が必要時にそれをアクセスできる。

【００２４】呼び出されるregisterService()メソッドの選択は、１）総称サービスが必要とされるか否か（すなわち、名前だけで識別可能なサービス）、或いは更にカスタム・サービスが必要とされるか否か（例えば名前同様、属性のリストにより識別されるサービス）、及び２）サービスが共用されるか否かに依存する。サービスが共用されない場合、サービスを登録するＡＰはクッキーへのアクセスを獲得するサービスだけである。それ以外では、サービスが存在すればクッキーが戻される。これはサービスに対して他のＡＰに与えられるクッキーと同じである。共用されるサービスがまだＶＥＭ内で使用可能でない場合、shared_service変数が、新たな共用サービスの可用性を反映するように更新され、新たなクッキーがＡＰに戻される。

【００２５】共用サービスへのアクセスは、アクセス制御リストを用いて制御され得る。代わりに、共用サービスが大域的に使用可能である、すなわち任意のＡＰが単にサービスを求めることにより、そのサービスに対してクッキーを受信できてもよい。共用サービスは、クライアント上に存在するサーバ・スタブの数を最小化するのに有用である。これらはまた、あるＡＰにより引き起こされるサーバ振舞いの変化が、別のＡＰにより相互的に使用される場合にも有用である。

【００２６】共用サービスは様々な方法でＶＥＭから除去され得る。第１の実施例によれば、基準カウントが各サービスに対して保持される。この基準は、現在幾つの

ＡＰがサービスへのアクセスを有するかを示す。基準カウントが０に達するとき、サーバがＶＥＳにより除去される。別の実施例によれば、最初にサービスを取り込んだＡＰだけが、それを除去することを許可される。

【００２７】そのサービスを含むようにＶＥＭ状態が更新されると、ＡＰはgetServiceHandle()メソッドを発行することにより、ハンドルを獲得するか、或いはgetServiceInstance()メソッドを用いることにより、スタブを獲得できる。この時、ＡＰは自由に任意の種類の接続を形成し、サービスを使用することができる。ＶＥＭは、ＡＰ及びサービスが如何に通信するかを制限するように実現され得る。サービスがもはや必要とされないとき、ＡＰはRevokeService()メソッドを呼び出し、これがＶＥＭ状態を次のように更新する。すなわち、サービスがＡＰに固有な場合、サービスが除去される。サービスが共用サービスの場合、それは上述の規則に従い除去される。

【００２８】以上、本発明は好適な実施例により述べられてきたが、当業者には様々な変更及び改善が考案されよう。従って、ここで述べられた好適な実施例は範例として提供されたものであり、本発明を制限するものではないことを理解されたい。

【００２９】まとめとして、本発明の構成に関して、以下の事項を開示する。

【００３０】（１）ネットワーク・クライアント上で実行されるアプリケーションに、ネットワーク・サービスへのアクセスを提供する方法であって、前記アプリケーションがリモート・ネットワーク・サービスへのアクセスを要求する時を判断するステップと、前記アプリケーションが前記リモート・ネットワーク・サービスへのアクセスを要求したとの判断に応答して、前記アプリケーションにより要求された前記リモート・ネットワーク・サービスを突き止めるステップと、少なくとも１つのオブジェクトをネットワーク・ノードのメモリにダウンロードするステップと、を含み、前記アプリケーションが前記リモート・ネットワーク・サービスを要求時にアクセスすることを可能にする方法。

（２）前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・スタブをダウンロードするステップを含む、前記（１）記載の方法。

（３）前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・ハンドルをダウンロードするステップを含む、前記（１）記載の方法。

（４）前記ネットワーク・クライアントのユーザに、以前にアクセスが許可された使用可能なネットワーク・サービスのビジュアル表示を提供するステップを含む、前記（１）記載の方法。

（５）前記ネットワーク・クライアントがネットワーク・コンピュータである、前記（１）記載の方法。

（６）前記ネットワーク・ノードがＪＡＶＡ端末である、前記（１）記載の方法。

（７）前記ネットワーク・ノードがパーソナル・デジタル・アシスタント（ＰＤＡ）である、前記（１）記載の方法。

（８）前記ネットワーク・サービスがシステム・サービスである、前記（１）記載の方法。

（９）前記システム・サービスが印刷サービスである、前記（８）記載の方法。

（１０）前記システム・サービスが永久記憶である、前記（８）記載の方法。

（１１）前記システム・サービスがシステム・モニタリングである、前記（８）記載の方法。

（１２）前記システム・サービスがシステム管理である、前記（８）記載の方法。

（１３）仮想環境マネージャが前記ネットワーク・サービスへのアクセスを提供する、前記（１）記載の方法。

（１４）前記仮想環境マネージャが、前記ネットワーク・ノード上で実体化される仮想マシン上で実行される、前記（１３）記載の方法。

（１５）前記ネットワーク・サービスへのアクセスがもはや必要とされない時を判断し、前記ネットワーク・ノードの前記メモリから前記オブジェクトの除去を開始するステップを含む、前記（１）記載の方法。

（１６）前記リモート・サービス・プロバイダにより、前記ネットワーク・クライアントが使用可能なサービスのセットを、第３者ディレクトリに登録するステップを含み、前記オブジェクトが前記ディレクトリを参照して、前記サービスへのアクセスを獲得する、前記（１）記載の方法。

（１７）前記ネットワーク・ノード上で実行される複数のアプリケーションにより、前記サービスへのアクセスを共用するステップを含む、前記（１）記載の方法。

（１８）あるアプリケーションにより必要とされるサービスへのアクセスを、前記アプリケーションにのみ知れる固有のキーによってのみ提供することにより、前記サービスを他のアプリケーションから分離するステップを含む、前記（１）記載の方法。

（１９）ネットワーク・コンピュータ上で実行されるアプリケーションに、システム・サービスへのオンデマンド・アクセスを提供する方法であって、

ａ）ユーザからの要求に応答して、リモート・ネットワーク・サービスへのアクセスを提供する仮想環境マネージャをダウンロードするステップと、

ｂ）システム・サービスのセットを識別するステップと、

ｃ）適切なリモート・サービス・プロバイダに連絡し、各前記サービスに対するクライアント・スタブまたはハンドルをダウンロードすることにより、前記システム・

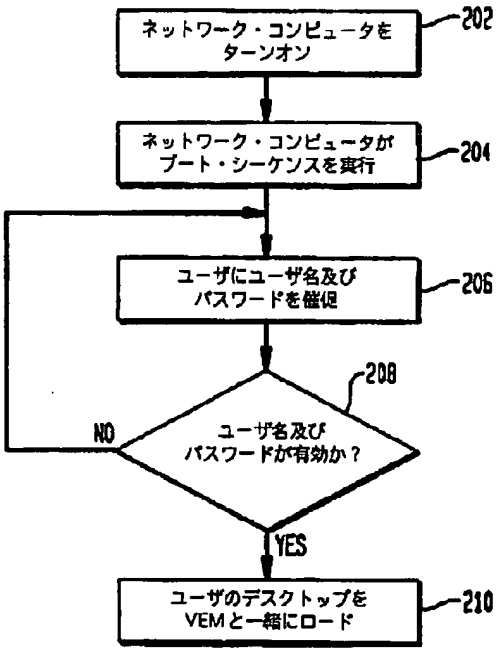
15

サービスのセットを構成するステップと、
ｄ）構成済みサービスのリストを保持するステップと、
ｅ）前記構成済みサービスへのアクセスを提供するステップと、を含む、方法。
（２０）複数のアプリケーションにより、前記システム・サービスへのアクセスを共用するステップを含む、前記（１９）記載の方法。
【図面の簡単な説明】
【図１】本発明と共に用いるのに好適な疎結合のシステムを示す図である。
【図２】ネットワーク・コンピュータのブート段階のフローチャートを示す図である。
【図３】各サービス・プロバイダにより実行される初期ステップのフローチャートを示す図である。
【図４】ＶＥＭが構成されたシステム状態の例を示す図である。
【図５】アプリケーションがネットワーク・サービスを使用しようとするときの、アプリケーションのフロー制御プロセスを示す図である。
【図６】デスクトップの例を示す図である。
【符号の説明】

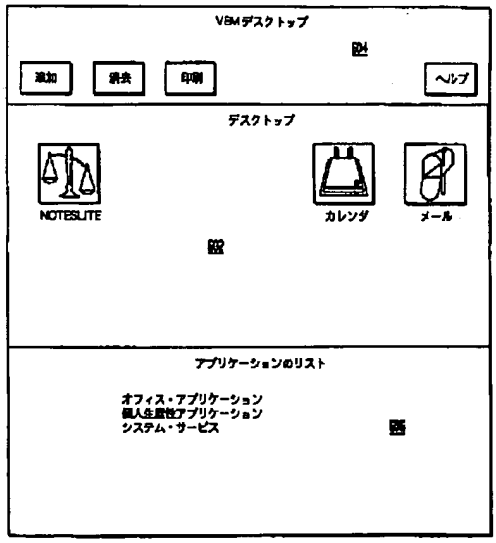
16

１０２　サービス・ディレクトリ・マネージャ（ＳＤＭ）
１０４（１）（２）（３）　サービス・プロバイダ
１０６（１）（２）、４１０　ネットワーク・コンピュータ
１０８（１）（２）　仮想環境マネージャ
１０９（１）（２）　仮想環境スーパバイザ（ＶＥＳ）
１１０（１）（２）　テーブル
１１２（１）（２）　通信ネットワーク
１１４（１）（２）　ウェブ・ブラウザ
４０２　プリント・サーバ
４０４　ファックス・サーバ
４０６　メール・サーバ
４０８　ＳＤＴ
４１２　アプリケーション
４１４　活動ＶＥＳオブジェクト
４１６　受動スタブ・オブジェクト
６０２　アイコン
６０４　ボタン
６０６　テキスト・リスト

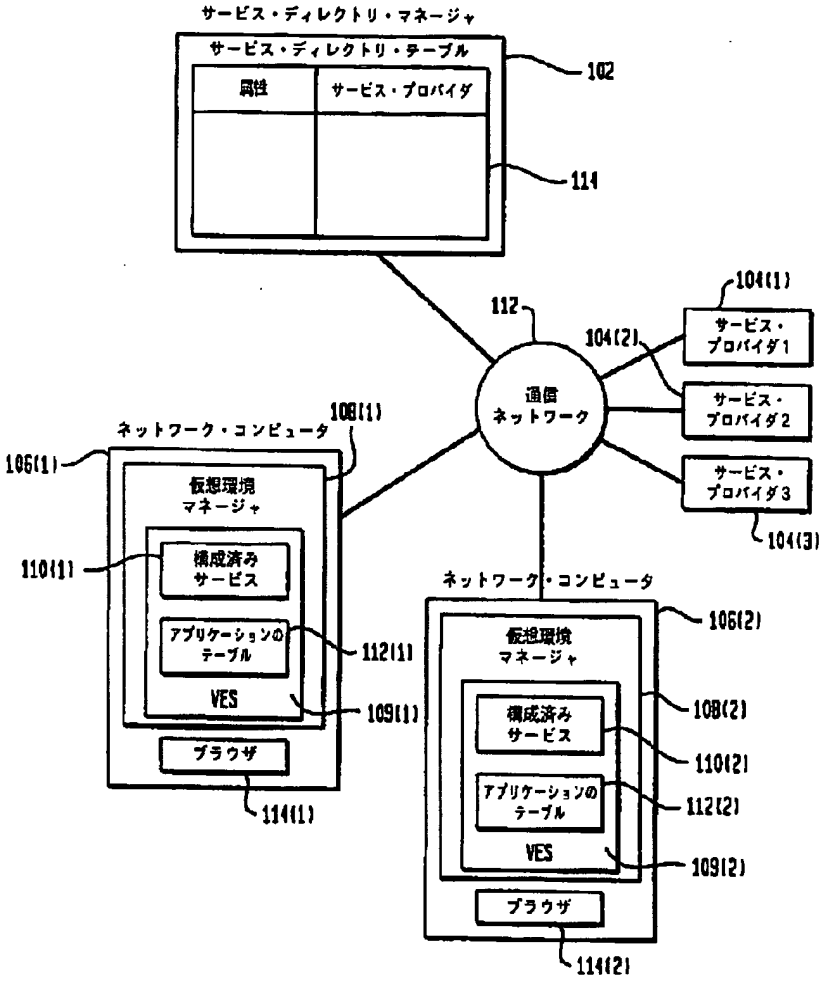
【図２】



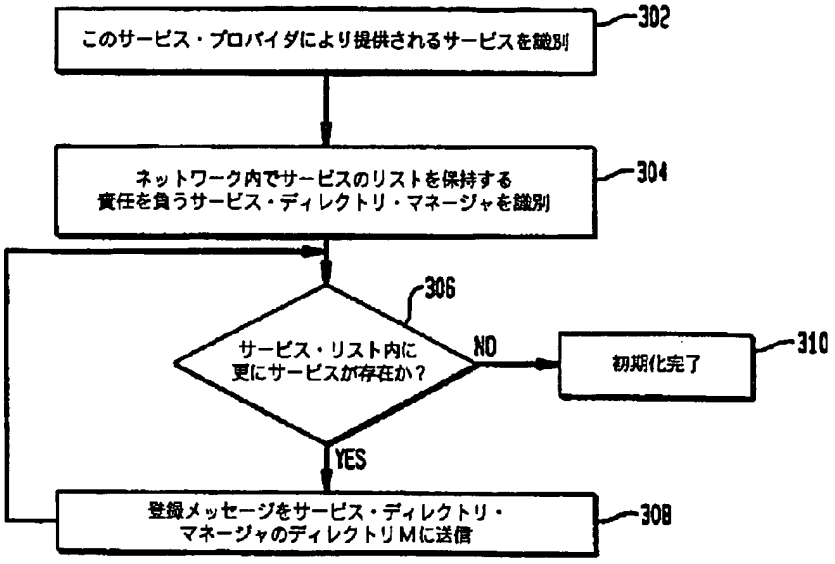
【図６】



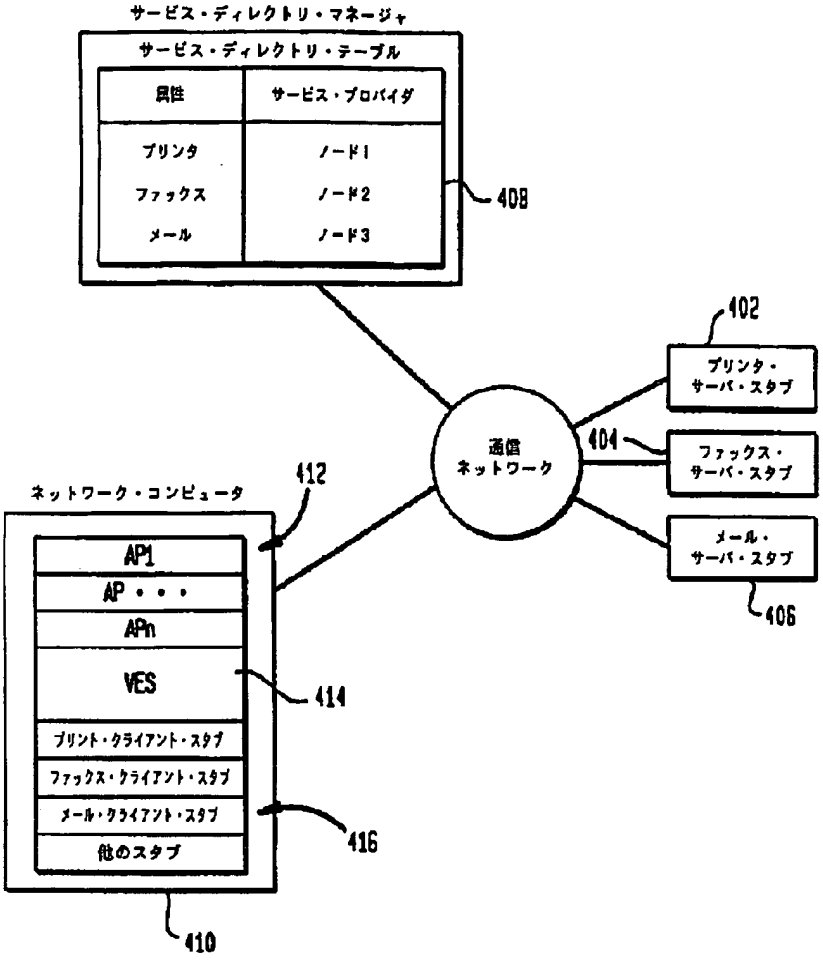
【図１】


サービス・ディレクトリ・マネージャ
サービス・ディレクトリ・テーブル
属性
サービス・プロバイダ
102
114
112
104(1)
104(2)
サービス・プロバイダ1
サービス・プロバイダ2
サービス・プロバイダ3
104(3)
通信ネットワーク
ネットワーク・コンピュータ
108(1)
106(1)
仮想環境マネージャ
構成済みサービス
110(1)
アプリケーションのテーブル
112(1)
VES
109(1)
ブラウザ
114(1)
ネットワーク・コンピュータ
106(2)
仮想環境マネージャ
108(2)
構成済みサービス
110(2)
アプリケーションのテーブル
112(2)
VES
109(2)
ブラウザ
114(2)

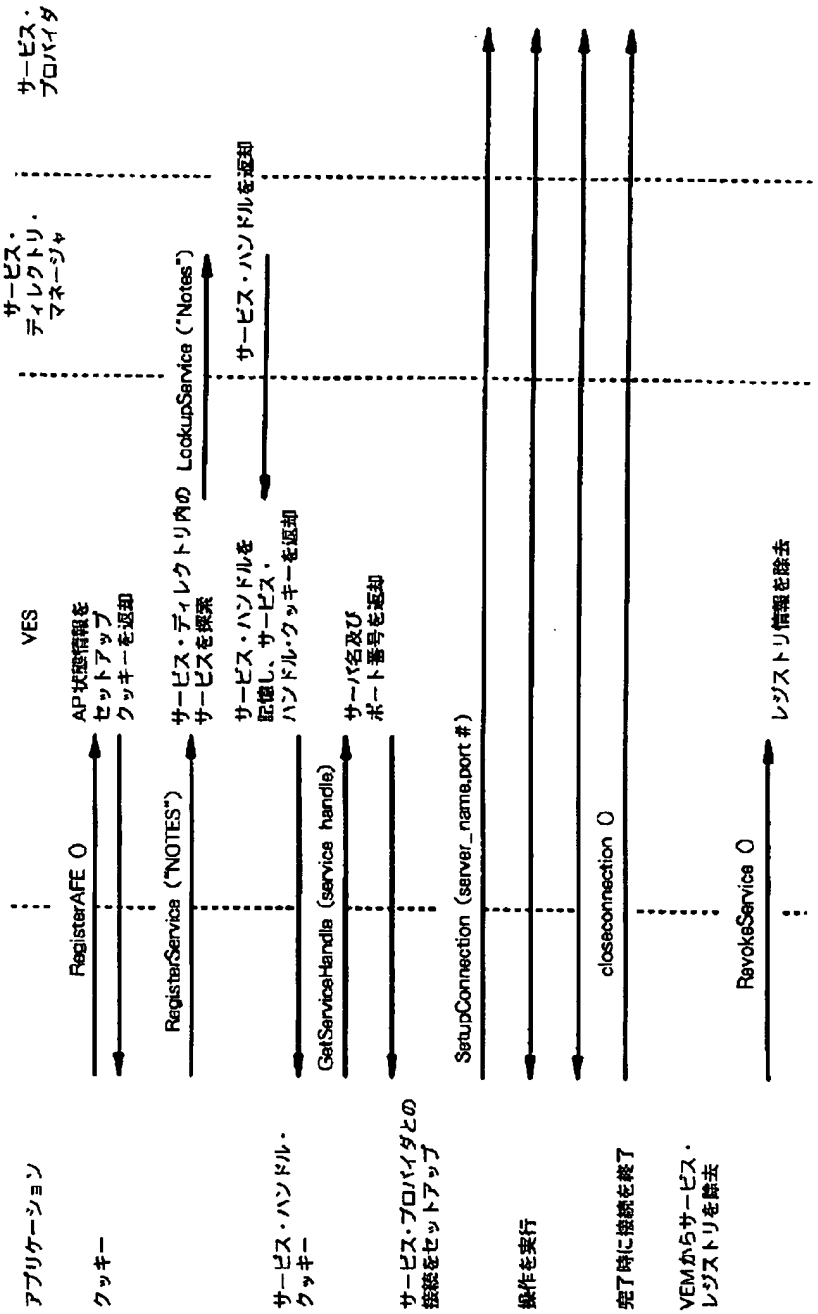
【図３】


このサービス・プロバイダにより提供されるサービスを識別
302
ネットワーク内でサービスのリストを保持する
責任を負うサービス・ディレクトリ・マネージャを識別
304
306
サービス・リスト内に
更にサービスが存在か？
NO
初期化完了
310
YES
登録メッセージをサービス・ディレクトリ・
マネージャのディレクトリMに送信
308

【図４】


サービス・ディレクトリ・マネージャ
サービス・ディレクトリ・テーブル
属性
サービス・プロバイダ
プリンタ
ファックス
メール
ノード1
ノード2
ノード3
408
402
プリンタ・サーバ・スタブ
404
ファックス・サーバ・スタブ
メール・サーバ・スタブ
406
通信ネットワーク
412
ネットワーク・コンピュータ
AP1
AP・・・
APn
VES
414
プリント・クライアント・スタブ
ファックス・クライアント・スタブ
メール・クライアント・スタブ
他のスタブ
416
410

【図５】



フロントページの続き

(72)発明者　アジャイ・モーインドラ
アメリカ合衆国10598、ニューヨーク州ヨークタウン・ハイツ、リン・コート　1340

(72)発明者　デボラ・ジーン・ズコウスキー
アメリカ合衆国10598、ニューヨーク州ヨークタウン・ハイツ、ビーバー・ドライブ　3211